

Analysing BrightTV data on SNOWFLAKE

1. Data Provided

a. Viewership table

Query: `select * from viewership;`

#	USERID	CHANNEL2	DURATION_2	RECORDDATE
1	903994	Trace TV	00:02:00	2016-01-04 20:09:00.000
2	802352	ICC Cricket World Cup 2011	00:10:06	2016-03-30 11:04:00.000
3	866125	ICC Cricket World Cup 2011	00:00:42	2016-03-30 19:01:00.000
4	2425604	ICC Cricket World Cup 2011	00:00:10	2016-03-25 17:23:00.000
5	793571	SawSee	00:01:33	2016-02-02 19:18:00.000
6	793571	Boomerang	00:00:10	2016-03-18 18:56:00.000
7	793571	Channel O	00:17:15	2016-03-20 06:25:00.000
8	793571	Boomerang	00:01:13	2016-02-03 08:43:00.000
9	819248	Channel O	00:04:45	2016-03-31 12:01:00.000

Query Details: Query duration 990ms, Rows 10K, Query ID 01bc4067-0001-01c0-0...

b. User profile table

Query: `select * from user_profiles;`

#	USERID	GENDER	RACE	AGE	PROVINCE
1	34	male	white	55	Free State
2	39	male	black	54	Eastern Cape
3	63	male	black	51	Kwazulu Natal
4	90	male	coloured	49	Kwazulu Natal
5	97	male	coloured	48	Free State
6	104	male	black	47	Gauteng
7	107	female	black	47	Gauteng
8	108	male	coloured	47	Western Cape
9	120	male	coloured	46	Western Cape

Query Details: Query duration 718ms, Rows 4.5K, Query ID 01bc406c-0001-01ad-0...

2. Combining the two tables

I used the **full outer join** function to combine the viewership and user_profile tables to return all the records in both tables. Then, renamed the combined table **Final_table**.

Query: `select A.Userid,gender, race, age, province, channel2, duration_2, recorddate from user_profiles as A full outer join viewership as B on A.userid=B.userid;`

#	USERID	GENDER	RACE	AGE	PROVINCE	CHANNEL2	DURATION_2	RECORDDATE
1	903994	male	coloured	33	Western Cape	Trace TV	00:02:00	2016-01-04 20:09:00.000
2	null	null	null	null	null	ICC Cricket World	00:10:06	2016-03-30 11:04:00.000
3	866125	male	Indian_asian	30	Gauteng	ICC Cricket World	00:00:42	2016-03-30 19:01:00.000
4	2425604	female	white	51	Gauteng	ICC Cricket World	00:00:10	2016-03-25 17:23:00.000
5	793571	female	white	9	Eastern Cape	SawSee	00:01:33	2016-02-02 19:18:00.000
6	793571	female	white	9	Eastern Cape	Boomerang	00:00:10	2016-03-18 18:56:00.000
7	793571	female	white	9	Eastern Cape	Channel O	00:17:15	2016-03-20 06:25:00.000
8	793571	female	white	9	Eastern Cape	Boomerang	00:01:13	2016-02-03 08:43:00.000
9	null	null	null	null	null	Channel O	00:04:45	2016-03-31 12:01:00.000

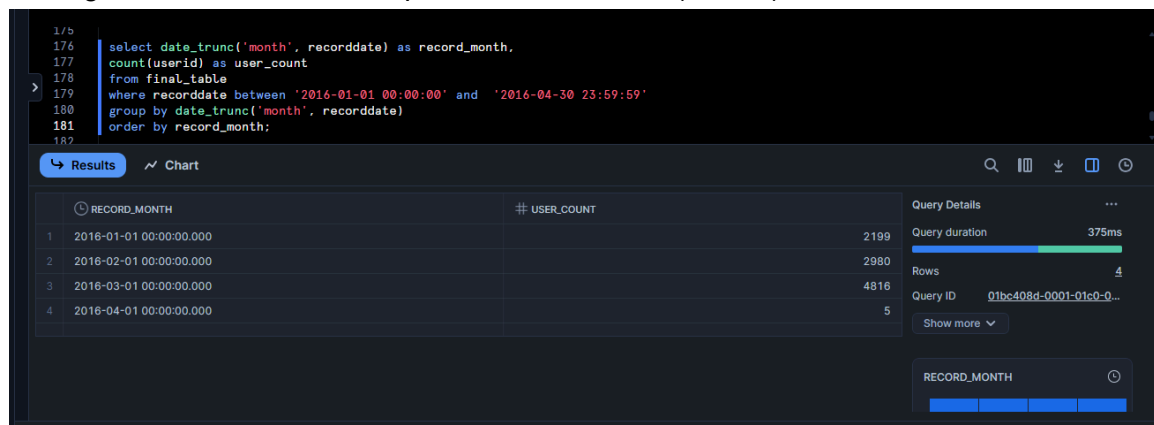
Query Details: Query duration 78ms, Rows 10.2K, Query ID 01bc4074-0001-0118-0...

3. Analysing data

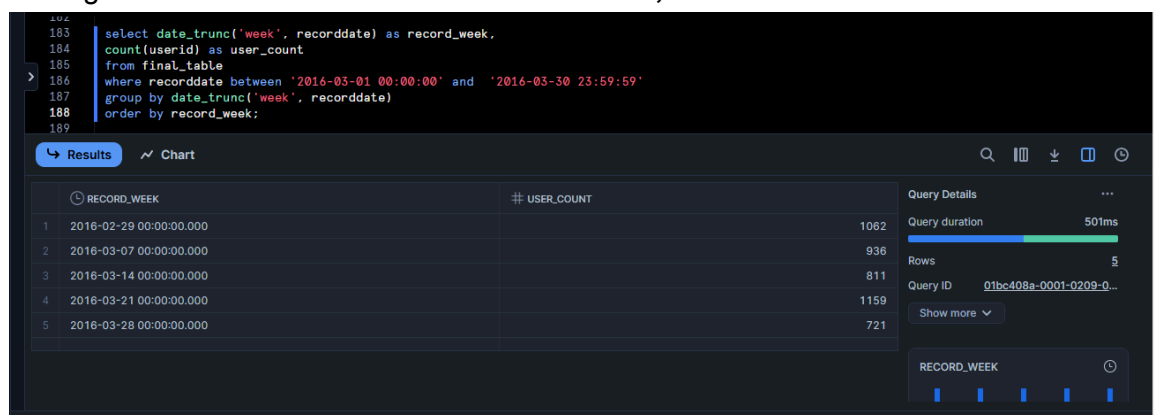
- a. Getting the **Average viewing time** using the **avg()** aggregate function. Also, converted the time to minutes.



- b. Getting the number of viewers per month. Month 3 (**March**) has more viewers.



- c. Getting the number of viewers in weeks for **March**, since it had more viewers.



- d. Counting the number of viewers per channel using the **count()** aggregate function.

```

172 select channel2, count(userid)
173 from final_table
174 group by channel2;
175

```

CHANNEL2	COUNT(USERID)
Trace TV	952
ICC Cricket World Cup 2011	1465
SawSee	251
Boomerang	714
Cartoon Network	793
Supersport Live Events	1638
SuperSport Blitz	896
E! Entertainment	367

Query Details: Query duration 64ms, Rows 22, Query ID 01bc4091-0001-01b2-0...

- e. Using a **CASE statement** to group age and **count** the number of viewers based on race.

```

144 select race, age, count(userid),
145 case
146 when age between 5 and 12 then 'school_age_children'
147 when age between 13 and 19 then 'teenagers'
148 when age between 20 and 35 then 'young_adults'
149 when age between 36 and 64 then 'adults'
150 when age > 64 then 'elderly'
151 end as Age_groups
152 from final_table
153 group by race, age;

```

RACE	AGE	COUNT(USERID)	AGE_GROUPS
coloured	33	60	young_adults
None	0	785	null
indian_asian	30	71	young_adults
white	9	10	school_age_children
indian_asian	34	78	young_adults
black	26	168	young_adults

Query Details: Query duration 95ms, Rows 323, Query ID 01bc409a-0001-0118-0...

- f. Figuring out the youngest and oldest viewers by using the **min()**(youngest) and **max()** (oldest) aggregate functions.

```

132 select min(age), max(age)
133 from final_table
134 where age != 0;
135
136

```

MIN(AGE)	MAX(AGE)
9	114

Query Details: Query duration 71ms, Rows 1, Query ID 01bc409e-0001-0208-0...

- g. Getting the total number of viewers using the **count()** aggregate function.

```

204 select count(userid) as total_viewers
205 from final_table;
206
207
208

```

TOTAL_VIEWERS
10989

Query Details: Query duration 174ms, Rows 1, Query ID 01bc40a5-0001-01ad-0...